TablesAndMetrics

November 30, 2020

1 Tables and Metrics

In this file, we generate some useful data for tables and statistical significance tests. We also generate PSNR/SSIM values for interpolation techniques in order to compare the results with those of the RFDN and RFDN1 networks. We also generate individual PSNR and SSIM values in order to compare between trained networks. Furthermore, this file was used to gauge the relative performances of the networks and analyze each networks performance.

Although this file was used for guaging the relative performance of trained models, **the main purpose of this file is to generate the tables and metrics for the report in an easy to read manner**. Other notebook files were used for model training, evaluation, and tuning.

[1]:
```
# Import modules
from interpolation import Interpolate
from evaluate import Evaluate
from RFDN import RFDN, RFDN1
import numpy as np
from compare import t_test, Compare
```

2 Preliminaries

Here, we set up some variables to be used throughout the notebook

[2]:
```
model1 = RFDN1(nf=10, upscale=2)
model2 = RFDN(nf=10, upscale=2)

data_dir = "/home/samuel/Documents/CMPUT511/Project/Data"

checkpoint_dir1 = "/home/samuel/Documents/CMPUT511/Project/Checkpoints/
 ↪AvgLearningCurve/RFDN1"
checkpoint_dir2 = "/home/samuel/Documents/CMPUT511/Project/Checkpoints/
 ↪AvgLearningCurve/RFDN"

checkpoint_file1 = "/home/samuel/Documents/CMPUT511/Project/Checkpoints/
 ↪AvgLearningCurve/RFDN1/checkpoint_2_40.tar"
checkpoint_file2 = "/home/samuel/Documents/CMPUT511/Project/Checkpoints/
 ↪AvgLearningCurve/RFDN/checkpoint_2_40.tar"
```

3 Calculate PSNR and SSIM for Image Interpolations

Here, we calculate the PSNR and SSIM measures for the traditional linearly interpolated upscaled versions of LR images. Notice that the PSNR and SSIM values are significantly lower than those produced by any of the trained networks, indicating that all the networks are better than traditional techniques.

[3]:
```
interp = Interpolate(data_dir)

psnr, ssim_ = interp.calculate_values()

avg_psnr = np.mean(psnr)
avg_ssim = np.mean(ssim_)

print(f"Average PSNR: {avg_psnr}")
print(f"Average SSIM: {avg_ssim}")
```

```
100%|      | 100/100 [02:10<00:00,  1.30s/it]Average PSNR: 6.613131931362553
Average SSIM: 0.027344164001678554
```

4 Evaluation metrics

Here, we show evaluation metrics such as average PSNR, SSIM, Inference Time, and Number of Parameters for different networks. These values are useful in gauging the relative performance of each of the networks.

[4]:
```
e = Evaluate(model1, checkpoint_file1, data_dir)
```

[5]:
```
values = e.get_values()
psnr = np.mean(values["psnr"])
ssim = np.mean(values["ssim"])
time = np.mean(values["times"])
params = sum(param.numel() for param in model1.parameters())
print(f"Average PSNR:              {psnr}")
print(f"Average SSIM:              {ssim}")
print(f"Average Inference Time:    {time}")
print(f"Number of parameters:      {params}")
```

```
100%|      | 100/100 [01:46<00:00,  1.07s/it]Average PSNR:
32.6316584421872
Average SSIM:              0.9209140539169312
Average Inference Time:    0.09533118963241577
Number of parameters:      9652
```

5 Statistical Significance

Here, we run a simple T-test for statistical significance for the RFDN and RFDN1 networks to determine if their average performance really is different.

[6]:
```
p_psnr = t_test(checkpoint_dir1, checkpoint_dir2, "psnr")
p_ssim = t_test(checkpoint_dir1, checkpoint_dir2, "ssim")

print(f"P-value for PSNR: {p_psnr}")
print(f"P-value for SSIM: {p_ssim}")
```

```
P-value for PSNR: 0.15422987536059868
P-value for SSIM: 0.24082446630607615
P-value for Loss: 0.05905154873981436
```

6 PSNR and SSIM Values for Comparing Two Networks

Here, we print out the mean PSNR, SSIM, and inference times for two networks in order to make a comparison of the two networks. We require very low inference times, while ensuring the PSNR and SSIM values are sufficiently high.

[7]:
```
comp = Compare(model1, model2, checkpoint_file1, checkpoint_file2, data_dir)
```

[8]:
```
values = comp.get_values()

psnr1 = np.mean(values["psnr"]["model1"])
psnr2 = np.mean(values["psnr"]["model2"])
ssim1 = np.mean(values["ssim"]["model1"])
ssim2 = np.mean(values["ssim"]["model2"])
times1 = np.mean(values["times"]["model1"])
times2 = np.mean(values["times"]["model2"])
```

```
100%|      | 100/100 [04:32<00:00,  2.73s/it]
```

[9]:
```
print(f"Average inference PSNR for model 1 ({str(model1)}): {psnr1}")
print(f"Average inference PSNR for model 2 ({str(model1)}): {psnr2}\n")
print(f"Average inference SSIM for model 1 ({str(model1)}): {ssim1}")
print(f"Average inference SSIM for model 2 ({str(model1)}): {ssim2}\n")
print(f"Average inference time for model 1 ({str(model1)}): {times1}")
print(f"Average inference time for model 2 ({str(model1)}): {times2}")
```

```
Average inference PSNR for model 1 (RFDN1): 32.6316584421872
Average inference PSNR for model 2 (RFDN1): 33.41725744285075

Average inference SSIM for model 1 (RFDN1): 0.9209140539169312
Average inference SSIM for model 2 (RFDN1): 0.927844226360321
```

```
Average inference time for model 1 (RFDN1): 0.0988073468208313
Average inference time for model 2 (RFDN1): 0.1525200605392456
```